Hindawi
Security and Communication Networks
Volume 2021, Article ID 9988646, 14 pages
https://doi.org/10.1155/2021/9988646

Research Article

BEvote: Bitcoin-Enabled E-Voting Scheme with Anonymity and Robustness

Ning Lu,[1,2] Xin Xu,[1] Chang Choi,[3] Tianlong Fei,[1] and Wenbo Shi[1]

[1]*College of Computer Science and Engineering, Northeastern University, Shenyang, China*
[2]*School of Computer Science and Technology, Xidian University, Xi'an, China*
[3]*Department of Computer Engineering, Gachon University, Seongnam 13120, Republic of Korea*

Correspondence should be addressed to Chang Choi; changchoi@gachon.ac.kr

Received 18 September 2021; Revised 8 November 2021; Accepted 24 November 2021; Published 15 December 2021

Academic Editor: Honghao Gao

When building the large-scale distributed decision control system based on mobile terminal devices (MTDs), electronic voting (E-voting) is a necessary technique to settle the dispute among parties. Due to the inherent insecurity of Internet, it is difficult for E-voting to attain complete fairness and robustness. In this study, we argue that Bitcoin blockchain offers better options for a more practical E-voting. We first present a coin mixing-based E-voting system model, which can cut off the relationship between the voter's real identity and its Bitcoin address to achieve strong anonymity. Moreover, we devise a secret sharing-based E-voting protocol, which can prevent voting number from being leaked ahead and further realize strong robustness. We establish the probable security theory to prove its security. In addition, we use the experimental evaluation to demonstrate its efficiency.

1. Introduction

1.1. Background and Motivation. Over the past two years, the rapid development of mobile edge computing (MEC) gives mobile terminal devices (MTDs) more computation and communication resource [1, 2], which produces the prospect of large-scale distributed decision control system based on MTD. For example, with the portable service of MTD, the medical intelligent diagnosis system allows multiple medical teams to carry out the consultation at any time and any place, which can effectively alleviate the burden of regional healthcare system caused by the emergent disaster (e.g., COVID-19); as MTD becomes more widespread, the intelligent property management system encourages more property owners to directly participate in community affairs, which facilitates the reduction in management costs, as well as the enhancement of management quality. It is widely known that in the distributed decision control system, settling the dispute among parties is an essential task. For example, in the multidisciplinary team (MDT)-based diagnosis treatment model, doctors usually have a disagreement in treatment plans for patients; in property management, different owners usually have different preferences, and among them, it is difficult to reach an agreement. In this case, electronic voting (E-voting), as a common means of settling such dispute (i.e., collecting the wish of each individual to evaluate group overall preference), is a necessary technique when building the large-scale distributed decision control system.

To improve the decision quality, besides the correctness of ballot counts, E-voting still needs to ensure the secrecy of ballots and protect the identity privacy of voters. Yet, due to the inherent insecurity of the Internet, it is easy for attackers to undermine the fairness of E-voting and leak the voters' identity privacy. For example, at DEFCON 25, hackers can easily invade the voting system to tamper with the data [3]. This would pose a serious security risk to the whole distributed decision control system. The advent of blockchain, however, offers a new appealing option to support E-voting service over the Internet [4]. It provides an opportunity to design an E-voting system that satisfies the above requirements. For instance, the traceability of each transaction in blockchain increases the feasibility of the verifiability for voting results; with the transparency of transactions,

blockchain-based E-voting would be equipped with the open-audit ability; the tamper-resistant nature of blockchain can effectively improve the voter confidence of voting; the pay-per-use nature of blockchain service provides incentives to encourage organizers to deploy voting service [5, 6]. It is well known that Bitcoin, the most widely used electronic cryptocurrency, owes its success to the fact that its online trading platform is built on blockchain. In this, to help lower costs, migrating traditional voting solutions to the Bitcoin platform becomes more of a natural and practical choice.

There are, however, two challenges in the implementation of Bitcoin-enabled E-voting, particularly if we also take into consideration the openness of decision control system and the constantly evolving threat landscape.

(i) *Anonymity*. To eavesdrop on the identity privacy of voters and further change their decisions to affect the fairness of the vote, the attackers may guess the true identities of transaction parties in blockchain. Bitcoin allows the input or output address that is not associated with user's real identity to conduct a transaction, but the sophisticated hackers can still infer their identities through clustering analysis of the transaction data. In this, Bitcoin is actually pseudo-anonymous. To guarantee the anonymity of the voter during voting process, although some relevant schemes based on zero-knowledge proofs or double envelope encryption have been proposed [7, 8], identity privacy can still be leaked through analyzing the voter-ballot relationship, once the voting result is announced. Therefore, how to enhance anonymity to ensure fairness becomes our first technical challenge.

(ii) *Robustness*. To pursue their own self-interest, even at the expense of others, the unscrupulous candidate would bypass the E-voting system to influence the voting result. For example, to illegally gain more voting rights, stakeholders can forge numerous virtual voters; to induce voters to make nonobjective judgments, attackers may eavesdrop and expose the ballots that any candidate has received during the voting process. Recently, there are more researches for E-voting, but research on its robustness has long been neglected. For example, Jason et al. utilize the prepaid Bitcoin cards to serve as voting rights, but do not give a secure way to distribute them [9]; Bistarelli et al. introduce tokens as ballots, which can be straightly voted for expected candidates at the cost of the leakage of voting process information [10]. Therefore, E-voting for decision control system includes a mechanism to resist such misbehaved candidate, which becomes our second technical challenge.

1.2. Proposed Scheme. To provide a better electronic voting service and address the above challenges, we propose a more secure Bitcoin-enabled E-voting scheme termed as BEvote. In contrast to existing similar work, BEvote attempt to implement the functions is shown in Table 1. In particular, in our scheme, we introduce a coin mixing-based system model, in which Bitcoin is viewed as ballot and the buyer and the seller in a transaction, respectively, represent the candidate and the voter. To achieve strong anonymity, we use coin mixing technique to cut off the relationship between the voter's real identity and its Bitcoin address before voted, and meantime, this anonymous address would be used in the entire voting process. To reduce the cost and improve the credibility, we use the Bitcoin blockchain to store voting data and execute the preset condition. To keep the voting numbers private during the whole voting process and thus achieve strong robustness, we adopt the joint Shamir random secret sharing technique, which can render the voting rights to be safely distributed to each voter.

The contributions of this study are regarded as the following to be threefold:

(i) The proposed BEvote abstracts a Bitcoin transaction as the voting process. Meantime, coin mixing is used to separate Bitcoin address from the real identity of voter, to achieve strong anonymity.

(ii) We devise a secret sharing-based E-voting protocol to realize the secure distribution of voting rights, which can prevent voting number from being leaked ahead.

(iii) We use security analysis to prove BEvote's anonymity and robustness. In addition, we conduct comprehensive simulations to demonstrate its efficiency.

The remainder of this study is organized as follows. Section 2 introduces relevant background materials and the related literature. Section 3 depicts the overall design of our scheme and then points out what specific threats it would face. Section 4 proposes a secure E-voting protocol to solve them. Sections 5 and 6, respectively, carry out security analysis and simulation evaluations. Section 7 summarizes the study.

2. Background and Relevant Literature

2.1. Electronic Voting Problem definition. To make all parties reach a consensus on important issues and take action to maintain consistency, electronic voting (E-voting) follows the principle that the minority is subordinate to the majority and derives the results based on the choices of voters. Assume that there are m candidates $\widetilde{C} = \{C_1, C_2, \ldots, C_m\}$ and n voters $\widetilde{U} = \{U_1, U_2, \ldots, U_n\}$. E-Voting needs to implement the function of n voters to choose one winner anonymously among m candidates, where the criteria for this winner are to gain at least k ballots. The voting procedure $V(\widetilde{C}, \widetilde{U})$ can be formalized as follows:

$$V(\widetilde{C}, \widetilde{U}) = f_1(\widetilde{U}) \cdot f_2(\widetilde{C}), \tag{1}$$

where $f_1(\widetilde{U})$ is used to vote on the candidates and $f_2(\widetilde{C})$ is used to collect ballots and generate statistics on them. Given $C_i \in \widetilde{C}$, when $f_2(C_i) > k$, we have the winner C_i.

Table 1: Function comparison between BEvote and previous work.

	[7]	[8]	[9]	[10]	[11]	[12]	BEvote
Strong anonymity	N	N	N	Y	Y	Y	Y
Robustness	Y	Y	N	N	Y	N	Y

(i) *Anonymity*. To protect the voters' safety, their identities should never be revealed throughout the voting stage.

(ii) *Validity*. In the event that adversaries tend to disguise their identity as system roles (e.g., voter or candidate) in attempts to participate in voting, there must be a mechanism to identify these forged roles.

(iii) *Robustness*. In any case, the voters cannot hold more voting rights than their own calculations would deem necessary. Thus, we need to prevent the Sybil attack and ballot forgery.

(iv) *Fairness*. To guard against bias, whether the voters or candidates, they cannot know ballot numbers before the end.

(v) *Receipt-free*. To prevent ballot trafficking, the bribed voters cannot prove to the candidate that they had the right to vote in advance.

(vi) *Verifiability*. Considering that the sophisticated attackers may tamper with the identity information of candidates, the entire voting process could be verified by any entity.

(vii) *Scalability*. With the scale of decision control system enlarging day by day, both voter numbers and candidate numbers should not be restricted by the performance issues.

2.2. Related Work. In recent years, as the distributed decision control system got more popular, the electronic voting problem has become a research hot spot. So far, several E-voting schemes have been proposed. For example, Chaum et al. proposed an E-voting protocol, which provided a secure transmission scheme for voting using Web applications such as email [13]; Adida et al. proposed an open-audit voting system that any willing observer can audit the entire process [14]; and Chondros et al. utilized a distributed subsystem to solve the single point of failure problem of E-voting [15]. In addition, Veronique et al. explained the verifiability notions of E-voting protocols [16]. Nevertheless, Candidates may bribe voters, which leads to collusion attacks, which is a sociological puzzle that cannot be technically solved. Benaloh et al. proposed the notion of receipt-free to comply with, which can prevent collusion to some extent [17]. Due to the lack of a trusted third party, the above schemes are difficult to realize the efficiency and fairness of voting.

Since the birth of blockchain technology, its anonymity and public verifiability have been proved to meet the need of the secure multiparty computation scenario [18]. Meantime, E-voting requires multiple parties to make a joint decision without revealing the privacy of voters and ballots, which is a typical secure multiparty computation scenario [19]. So far, there have been attempts to design the E-voting schemes based on blockchain platforms, including Ethereum and Zcash [20–23]. As is well known, the supreme status of Bitcoin blockchain is hard to shake, although other existing blockchain platforms have better technical characteristics in certain aspects. This is also the main reason why the upper layer protocols such as the Bitcoin lightning network that optimizes and enhances Bitcoin's performance and functions are still booming [24]. Therefore, this study mainly focuses on investing in Bitcoin-enabled E-voting schemes.

Depending on whether the coin mixing is used, existing Bitcoin-enabled E-voting protocols are classified as "with coin mixing" or "without coin mixing."

2.2.1. Bitcoin-Enabled E-Voting Schemes without Coin Mixing. Zhao et al. earlier utilized Bitcoin blockchain to design an E-voting protocol [7], which requires additional stages to distribute secret random numbers through the zero-knowledge proof algorithm zk-SNARKs. However, it can only distribute random numbers for two candidates, which results in the protocol only supporting 1-of-2 voting. Besides, all voters need to construct a reveal transaction before voting, which is exposed to the public. Because that associates the ballots with the voter, the candidate can know whether any voter has voted for him, which leads to the loss of anonymity. Jason et al. proposed a kind of card that contains prepaid Bitcoin addresses and corresponding private keys, in which regulators can place the PBC in an envelope when it is issued to ensure that it cannot be traced back to voters [9]. Nevertheless, this way of distributing physical objects runs counter to the original intention of E-voting to save manpower and material costs. Moreover, electronic virtual cards could be used, but failed to come up with a plan that would guarantee the anonymous distribution of cards. Furthermore, it cannot prevent dishonest staff from disclosing information related to cards and voters, and meet the requirements of voting anonymity. Adiputra et al. proposed to combine Bitcoin blockchain and the double envelope encryption technique for E-voting [8]. However, the public key to encrypt all the ballots generated by the electoral commission can only guarantee the anonymity of the voter during the voting process. Once the electoral commission releases the private key, the voter and the ballot would be associated. Thus, this scheme has serious anonymity issues. In addition, Bitcoins cannot be redeemed even if the vote fails, so the robustness is poor. Besides, the electronic commission equivalent to the supervisor of our protocol is responsible for too many tasks, which leads to the scheme being too centralized. Zhang et al. proposed the chaintegrity protocol. It utilizes a blind signature to ensure the anonymity and can only be used in permissioned chains without transaction fees [25]. The reason for this is that prepaid transaction fees can be associated with the user's identity. In addition, the so-called platform-independent feature is not established since it does not support permissionless blockchains, particularly Bitcoin. To sum up,

coin mixing is necessary for Bitcoin-enabled E-voting scheme.

2.2.2. Bitcoin-Enabled E-Voting Schemes with Coin Mixing. Bartolucci et al. designed a share-based blockchain voting scheme called SHARVOT [11], which utilizes the circle shuffle to obfuscate the relationship between the ballots and the voter's identity. However, considering that the size of the transaction data in the Bitcoin protocol is limited, the P2SH script corresponding to each unlock condition of the vote commitment transaction needs to contain all the ballots obtained by a candidate, which makes the multi-signature used in the P2SH script have only 15 public key positions. This means that, after excluding the two required public key positions, a maximum of 13 ballots can be placed, which leads to a $n < 13m$ limit between the maximum number of voters n and the number of candidates m. It also limits the maximum number of shares to reconstruct private key to 13. These restrictions make SHARVOT unable to meet the requirements of multiple candidates and multiple voters. Takabatake et al. adopted Zerocoin to assist Bitcoin to provide anonymity for voting, in which Zerocoin can be used to obtain a zero-knowledge commitment before it can be exchanged for a new Bitcoin address [12]. However, the Zerocoin blockchain is not compatible with Bitcoin. Using two kinds of blockchains would greatly increase the complexity of actual operations. In addition, the ballots are placed directly in the OP_RETURN field of the Bitcoin output script. All the ballots need to be checked to count the winners. The efficiency is lower when the voting scale is larger. Bistarelli et al. proposed an end-to-end Bitcoin voting platform, which utilizes anonymous Kerberos to conceal the relationship between the Voter's identity and the Bitcoin address, as well as uses tokens to represent the ballots [10]. Nevertheless, voters sending tokens directly to the candidates through a public Bitcoin transaction would cause severe consequences. Since all transactions cannot be packaged into new blocks by miners at the same time, the voters could know the current number of ballots of any candidate by the number of tokens received. The leakage of the voting situation tremendously damages the fairness of voting. In short, the above schemes are not applicable for the large-scale distributed decision control system due to their deficiency in anonymity and scalability.

3. Bitcoin-Enabled E-Voting Scheme

In this section, we introduce the system model and the functionality of system components. Afterwards, we present the potential threats faced by BEvote and defer their solutions in Section 3. Table 2 lists the notations in our paper.

3.1. Motivation and Assumptions. We exploit the Bitcoin blockchain platform for implementing voting. Both verifiability and scalability requirements are considered as the main limiting factors for electronic voting in large-scale distributed decision control system. However, over time, technology advances increase the feasibility of large-scale E-voting solution. Along with the spreading of blockchain in secure multiparty computation, researchers start to propose the blockchain-enabled E-voting, where the digital transaction process is abstracted as one voting. In traditional E-voting, it is hard to achieve high availability and open verifiability simultaneously. Yet, by virtue of the traceability in blockchain, they are readily available. Going a further step, to allow more people to participate in voting and verify results, Bitcoin blockchain, as a public trading platform with daily business transaction of up to 500 million USD, is an appealing option. In this, the scalability of E-voting could also be ensured. In addition, the pay-per-use nature of transaction service encourages the Bitcoin platform to support voting service. Consequently, it is not only technically sound but also economically preferable to migrate E-voting to Bitcoin blockchain platform.

Table 2: Notations used to describe the BEvote design.

Notation	Description
U_i	The Voters
C_j	The Candidates
v_i	Ballots of U_i
$(pu_i, pr)_i$	The public/private key pair of Voters
(PU_j, PR_j)	The public/private key pair of Candidates
(P, S)	The public/private key pair of secret sharing
s_i	The secret shares
p_i	The public key share of s_i
G	Basic point of elliptic curve
VoteTx_i	Voting transactions of U_i
WinTx_j	Winning transactions of C_j
RefundTx_i	Refund transactions of U_i
ABA_i	Anonymous Bitcoin addresses of U_i
AVL	Anonymous voters list
ID_{C_j}	Identity number of C_j
info	Identity information
Reg	Registration
Enc	Encryption
Sig	Signature
T	Timeout time

We make a few assumptions to assist BEvote's design, part of which have already been supported by the current blockchain technologies, while others can be satisfied through existing research:

(1) We assume that the blockchain generates new blocks at fixed intervals without forks.

(2) We assume that the data in the blockchain are always public auditable, while the entire transaction history can always be read. However, only if valid transactions are submitted and the consensus is obtained the blockchain can be written.

(3) We assume that no adversary can destroy the blockchain's proof of work consensus mechanism. This means that the data cannot be tampered with.

3.2. System Model. Although Bitcoin blockchain adopts the pseudonym mechanism that tries to achieve exchange anonymous security, the sophisticated attackers still alone utilize the simple analysis technology to analyze all

transactions that are transparently recorded in any one of the chain nodes, which could infer the true identities of transaction parties. In this case, utilizing Bitcoin blockchain directly cannot guarantee the strong anonymity of E-voting. To solve such an issue, coin mixing becomes the natural choice, providing a service that obfuscates the ownership of Bitcoin to interrupt the interlinkage of transaction parties. However, the traditional coin mixing usually engages a set of chain nodes as middlemen and then allows them to combine multiple transactions into one transaction, which significantly affects its scalability, considering Bitcoin's maximum transaction size. Apparently, it is not applicable for the large-scale E-voting system, because on one hand the smaller size anonymous set cannot support strong anonymity for E-voting and on the other hand the large computation overhead degrades the efficiency of voting. For this, we have proposed a scalable coin mixing scheme [26, 27], which breaks down the trading process into transferring stage and paying stage. The former deposits the transaction Bitcoins on a specified middleman (termed as holding Mixer) and further cuts off the link between these Bitcoins and the seller; the latter specify another middleman (termed as Paying Mixer) to advance the Bitcoins to buyer and then pay back the Bitcoins to this middleman after the mixing is over. In doing so, we can not only improve the size of anonymous set but also improve the execution efficiency, which can satisfy the requirement of large-scale mixing Bitcoins. Based on it, we embed such coin mixing into our E-voting scheme.

In Bitcoin-enabled E-voting, each voting can be abstracted as a transaction, where the voter and candidate can be viewed as the buyer and the seller, respectively. Moreover, the voter requires two Mixers to interrupt its interlinkage with the candidate before voting, which could facilitate concealing the voter's identity during the entire voting stage. Furthermore, motivated by superior returns, abundant chain nodes are willing to provide mixing service and serve as Mixers. Meantime, to achieve flexibility, any system role is allowed to enter or exit at any time. In this case, compared with decentralization, a completely centralized structure is more efficient, which could reduce the expense of management. Guided by these principles, as depicted in Figure 1, we devise a three-tiered system model based on coin mixing. In data persistence tier, all the status information related to voting (including the course and the result) would be recorded in Bitcoin blockchain; in business tier, Voters, Mixers, and Candidates, as partners, would be scheduled in a fixed order so as to realize the voting task; in management tier, all system roles should be centrally controlled with great care.

(i) *Supervisor* provides functionality for validating each requester's qualifications, including *Voter*, *Can di date*, and *Mixer*. It would maintain a member list for each system role. Once monitoring any bad behavior, it excludes this member from the corresponding list. In addition, it also provides *Mixer* recommendation service for *Voters*. Generally, it is authorized by the government.

(ii) *Voter* is the one that could obtain voting rights. This means that it can generate ballots and vote for its desired candidates.

(iii) *Candidate* is the one who is voted. When the polls close, the candidate with the most ballots wins.

(iv) *Mixers* provide functionality for concealing voters' identities, in which holding Mixer is used to host the voter's Bitcoins, and trading Mixer is used to transfer them to the specified candidate.

(v) *Bitcoin blockchain* is utilized as an infrastructure for data storage. It would record all information relating to the state of voting in an anonymous and tamper-resistant manner.

Under the ideal network environment, the workflow of BEvote can be stated as follows:

(i) *Step 1*. Registering: Each participant (including Voter, Mixer and Candidate) is required to submit registration request to Supervisor. Only successful registration can provide relevant functions.

(ii) *Step 2*. Confusing Voter's identity: Each Voter has two Bitcoin addresses: an original address that contains Bitcoins and a brand new address without Bitcoins. To mix its identity with others, it resorts to Mixer's mix coin strategy and further transfers Bitcoins from the original address to the new address without leaving a trace. Thus, the new one is anonymous.

(iii) *Step 3*. Generating ballots: Each Voter uses the public key of Supervisor to encrypt the information of Candidate to generate the ballot.

(iv) *Step 4*. Constructing voting transaction: Each Voter utilizes the scripts in the Bitcoin protocol to construct a voting transaction. There are two unlock conditions for the transaction output: the one associated with ballots is performed by the Supervisor; another is performed by the Voter itself after a timeout.

(v) *Step 5*. Collecting ballots: The Supervisor searches the voting transactions in the Bitcoin blockchain according to an anonymous Voter list, then decrypts the ballots with its private key, and counts them. As long as there is a Candidate who derives enough ballots before the timeout, the voting is success. Afterwards, Supervisor constructs the winning transaction to collect all the ballots of the winner. Meantime, those Voter without voting for the winner need to construct a refund transaction to unlock its corresponding transaction and redeem Bitcoins. Conversely, if no candidate wins, the voting fails. Each Voter unlocks its transaction and redeems coins.

3.3. Voter's Identity Confusion Strategy. To confuse its identity in voting, the Voter U_i needs to apply to both Supervisor and Mixers for the coin mixing service so as to

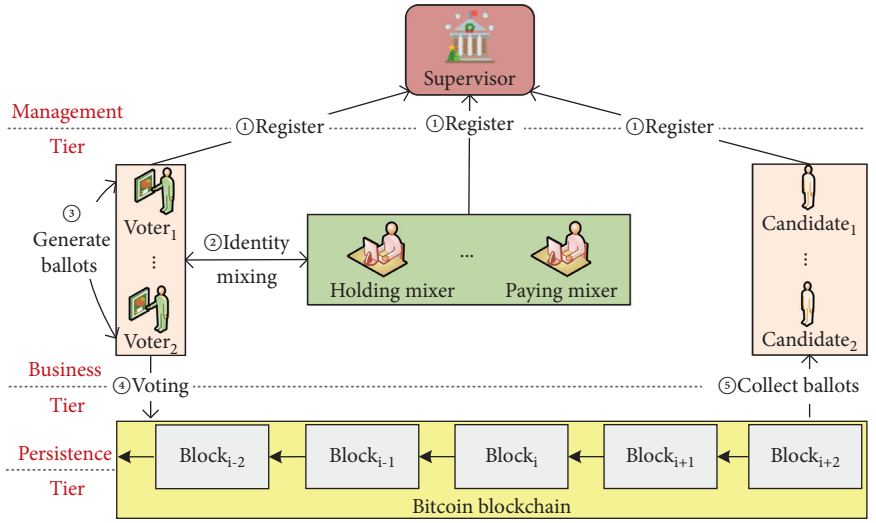

Figure 1: System model.

transfer its Bitcoins from original address OBA_i to anonymous address ABA_i without any tracks. As is depicted in Figure 2, the primary principle of coin mixing can be stated as follows: when receiving mixing request, Supervisor randomly select two Mixers with lower load and recommend them to U_i, in which one is the hosting Mixer and another is the paying Mixer. First, U_i constructs a transaction that transfers its Bitcoins from OBA_i to hosting Mixer's escrow address. Once confirming this transaction, the hosting Mixer needs to reply with a voucher. Second, U_i directly forwards this voucher to the paying Mixer and requires it to construct a new transaction that transfers the equal number of Bitcoins from this Mixer's private address to ABA_i. At last, Supervisor gives Bitcoins back to the paying Mixer so as to recover costs, after the task is finished. Looking at the whole process, we can easily find a vulnerability that U_i's identity privacy would still be exposed when any Mixer colludes with adversaries. In this, we introduce the group blind signature and utilize its unlinkability feature to cut off the interlinkage between these Mixers.

For each registered Mixer_i, Superviser would assign a group certificate $v_i = (y_i + 1)^{1/e} (\text{mod} n)$ and an escrow address to it, where e is a bilinear pair and y_i denotes the proof of ownership for Mixer_i's private address. Meantime, it releases the Mixer list $L = \{\text{Mixer}_0, \ldots, \text{Mixer}_m\}$ on the blockchain platform. In the following, we illustrate the group blind signature-based coin mixing in detail.

Step 1: After U_i has registered successfully, it immediately records the current time as T_1. When receiving the *Mixers* recommended by *Superviser*, it sends an confusion parameter message $M_1 = \langle T_4, T_5 \rangle$ to the paying *Mixer* on an anonymous channel, where T_4 is the deadline for U_i to provide the voucher and T_5 is the deadline for this *Mixer* to submit tx_3.

Step 2: Once the paying *Mixer* accepts the message M_1, it sends the message $M_2 = \langle \text{Sig}\{M_1, \text{nonce}\}_{xb}, M_1, \text{nonce} \rangle$ to U_i on the anonymous channel, where *nonce* is random number and x_b denotes the private key of the paying *Mixer*.

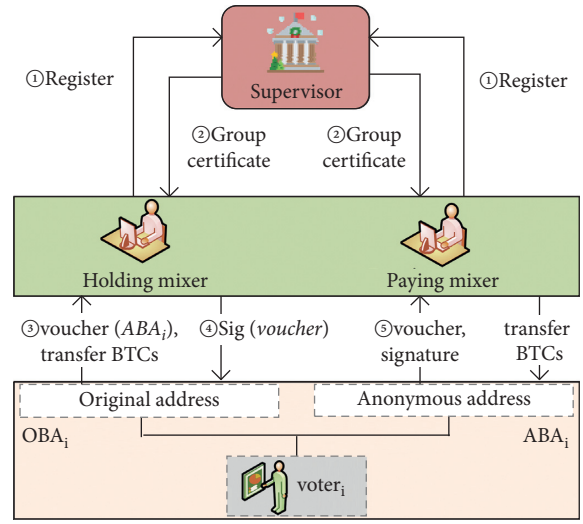

Figure 2: An example of identity confusion.

Step 3: In the meantime, U_i sends an confusion parameter message $M_3 = < T_2, T_3 >$ to the hosting *Mixer*, where T_2 is the deadline for U_i to submit the Bitcoin transaction tx_2 and T_3 is the deadline for U_i to execute group blind signature.

Step 4: Once the hosting Mixer accepts the message M_3, it sends $M_4 = \langle \text{Sig}\{M_3, \text{hosting address}, \text{nonce}\}, M_3, \text{hosting address}, \text{nonce} \rangle$ to U_i, where hosting address denotes the escrow address of the hosting Mixer and x_a denotes the private key of the hosting Mixer.

Step 5: When receiving M_4, U_i constructs the transaction tx_2: $U_i \longrightarrow$ hosting Mixer before the time point T_2 and transfers its Bitcoins from U_i's original address OBA_i to the escrow address of the hosting Mixer. Then, it sends a signatured message $M_5 = \langle \text{Sig}\{ID_{tx_2}, \text{Own}_v\}_{yv} \rangle$ by the private key of U_i to the hosting Mixer and expose it on the blockchain platform, where Own_v is the proof of

ownership of the U_i to OBA_i and y_b denotes the public key of the paying Mixer.

Step 6: When the hosting *Mixer* receives the message M_5 through blockchain platform, it needs to complete the following verifications: (1) whether the transaction amount of tx_2 is f, where f is the fixed denomination of the transaction, and (2) whether Own_v has not been used. Once M_5 is validated, the hosting *Mixer* executes the blind SKLOGLOG protocol and blind SKROOTLOG protocol before the time point T_3, and meantime, the entire execution process would expose on the blockchain platform. For the details, refer to Ref. [28]. Then, it sends message $M_6 = \langle (t_i^{SKLOGLOG} + eb), t_i^{SKROOTLOG}(af)^b \rangle$ to U_i and exposes it on the blockchain platform, where t_i is an arbitrary big value.

Step 7: After receiving the message M_6, U_i calculates $c_1 = \text{SKLOGLOG}_l[\alpha | z = g^{a\alpha}](m)$ and $c_2 = \text{SKROOTLOG}_l[\beta | z\ g = g^{\beta e}](m)$ and further obtains the group blind signature (g, z, qr_1, qr_2) of the voucher $\{ABA_i, \text{payingMixer}, \text{nonce}\}$, where qr_i is the quadratic residue in the referenced protocol, g is the generator of the cyclic group, a is the identity element of the cyclic group, and z is the group member key. Then, it sends a message $M_7 = \langle M_2, \{ABA_i, \text{paying Mixer}, \text{nonce}\}, \langle (g, z, c_1, c_2)$ to the paying Mixer and publishes it on the blockchain platform before time point T_4.

Step 8: When the paying Mixer receives the message M_7 through blockchain platform, it needs to complete the following verifications: (1) whether (g, z, c_1, c_2) is the group blind signature of $\{ABA_i, \text{paying Mixer}, \text{nonce}\}$, (2) whether M_7 has not been used, (3) whether M_2 is equal to the agreement signature in Step 2, and (4) whether M_2 has not been used. When the verifications are completed, the paying Mixer constructs the exchange transaction tx_3: paying Mixer $\longrightarrow U_i$ before the time point T_5 and transfers its Bitcoins from the private address of the paying Mixer to the U_i's anonymous address ABA_i.

Supervisor can regularly audit the confusion data recorded on Bitcoin blockchain platform to detect the malicious *Mixers* and punish them. For example, through auditing the hosting *Mixer*'s commitment and tx_2, the lazy behavior is detected; through auditing the voucher of the paying *Mixer*, the denial-of-service behavior is detected.

3.4. Voting Transaction Construction Strategy. In the Bitcoin blockchain platform, the script is the common technique to construct the voting transaction [29]. Generally speaking, each transaction can contain multiple input scripts and multiple output scripts, in which each input script is associated with the output script of the source transaction up to coinbase transaction. In BEvote, we adopt the P2PKH script and the $M - N$ combined P2SH script. The former could provide the Check Lock Time Verify (CLTV) operator. Its output script only stores the hash value of the public key, and input script stores the public key for verifying whether it matches the hash value. The latter uses a redeem script to store the public key of single or multiple signatures. Assume to have N public keys, and at least M of the public keys must be provided with a signature corresponding to the unspent transaction output. Its input script must give all the private keys and the serialized redeem scripts.

We use an example to illustrate the voting transaction construction. As is shown in Figure 3, there is a voting transaction $VoteTx_i$ with a ballot. Its output is x Bitcoins, and input address is the anonymous Bitcoin address ABA_i. The output script is based on an *OR* operator. This means that the transaction output can be unlocked only if either of the following conditions is satisfied.

Condition 1. is set to be performed by the *Supervisor*, and the ballot is recorded in the P2SH script. When creating an $M - N$ combined P2SH UTXO, the user first needs to generate a redeem script to store all public keys. In the Bitcoin protocol, the PUSHDATA operation of the stack-based language limits the maximum data to 520 bytes; hence, the redeem script can only place up to 15 public keys under this restriction. The script form is as follows:

$$M\langle \text{PubKey}_1 \rangle \langle \text{PubKey}_N \rangle N \text{ OP\_CHECKMULTISIG.} \tag{2}$$

Then, the HASH160 algorithm is used to generate a 20 byte long hash of this script and an output script is created as follows:

$$\text{OP\_HASH160 } \langle \text{Hash of Redeem Script} \rangle \text{OP\_EQUAL.} \tag{3}$$

Data can also be stored in the P2SH script, and the idle public key bits in the redeem script are just replaced with data. For example, if there are data $Da\ ta_1$, $Da\ ta_2$, the redemption script form is as follows:

$$M\langle \text{PubKey}_1 \rangle \langle \text{PubKey}_N \rangle \langle \text{Data}_1 \rangle \langle Da\ ta_2 \rangle N \text{ OP\_CHECKMULTISIG.} \tag{4}$$

In BEvote, the redeem script *RS* corresponding to the first unlock condition of $VoteTx_i$ output script contains the public key PU_s of *Supervisor* and stores the ballot v_i. It can be unlocked by the private key of *Supervisor*. The form of $1 - 1$ combination script is as follows:

$$1\langle PU_s \rangle \langle v_i \rangle 1 \text{ OP\_CHECKMULTISIG.} \tag{5}$$

The output script stores the 20 byte long hash value of *RS* calculated by HASH160 algorithm in the form:

$$\text{OP\_HASH160}\langle \text{Hash}(RS) \rangle \text{OP\_EQUAL.} \tag{6}$$

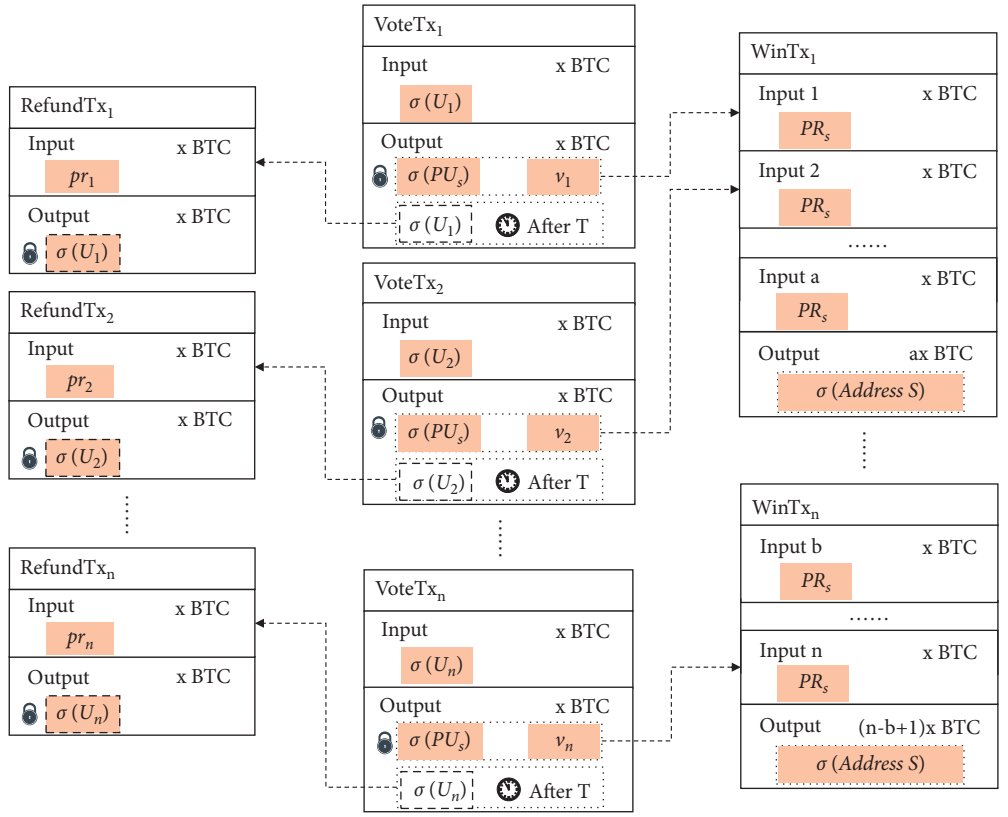

Figure 3: An example of voting transaction.

Condition 2. is added in the form of a time lock with the P2PKH script. CLTV can regularly lock an output of a transaction, instead of locking the entire transaction. After using the CLTV operator in the transaction output, it can limit the output to be unlocked only after a specified time. For example, Alice transfers to Bob, usually using an output script in the form of P2PKH:

$$\text{OP\_DUP OP\_HASH160}\langle\text{Hash of Bob}^{A}\text{s Public Key}\rangle\text{OP\_EQUALVERIFY OP\_CHECKSIG.} \tag{7}$$

CLTV is used to lock for a period of time; assuming 2 months, the time can be set to the current block height +8640 (block) or the current Unix epoch time +5184 000 (seconds). After the timeout, P2PKH transactions can be unlocked by Bob using his own private key signature:

$$\begin{aligned}&\langle\text{now} + 2\ \text{months}\rangle\\&\text{OP\_CHECKLOCKTIMEVERIFY OP\_DROP OP\_DUP OP\_HASH160}\\&\langle\text{Hash of Bob}'\text{s Public Key}\rangle\text{OP\_EQUALVERIFY OP\_CHECKSIG.}\end{aligned} \tag{8}$$

In BEvote, the P2PKH script of the second unlock condition of VoteTx_i output script contains the voter's public key pu, which is locked for a period of time using the CLTV operator. After the timeout, the voter can use his private key signature to unlock it. When locked for one day, the form is as follows:

$$\begin{aligned} &\langle \text{now} + 1\text{day} \rangle \\ &\text{OP\_CHECKLOCKTIMEVERIFY OP\_DROP OP\_DUP OP\_HASH160} \\ &\langle pu \rangle \text{OP\_EQUALVERIFY OP\_CHECKSIG.} \end{aligned} \tag{9}$$

3.5. Potential Threats. The above strawman design would face the following potential threats in real environments:

(i) Sybil attack: To illegally gain more voting rights, attacker may generate multiple Bitcoin addresses to forge *Voters*'s identities.

(ii) Premature leakage of ballots: With the opportunity of generating ballots, the misbehaved insider of *Supervisor* may expose the ballots when the voting has not yet been closed.

To fix these threats and enhance its robust, we devise a secure voting protocol in Section 4.

4. Secure E-Voting Protocol for BEvote

In this section, we describe a secure E-voting protocol to defend against the above threats in BEvote.

4.1. An Overview. To establish an efficient countermeasures, we need to analyze the causes of these threats. First, the openness of Bitcoin blockchain platform and its multi-address allocation makes it possible for hackers to forge numerous virtual *V oters*. In this case, increasing barriers to entry would be effective to defend against the Sybil attack. Thus, adding an authentication function in E-voting protocol is necessary. Second, to avoid premature leakage of ballots by *Supervisor*, the voting transaction only can be unlocked by those winning candidates; i.e., they at least gain k votes cast. Thus, it is necessary to design a feasible encryption policy for voting transaction.

Guiding by these principles, we first design an authentication based on public key cryptography to prevent the virtual *Voters*. Second, we introduce the joint Shamir random secret sharing (joint-RSS) [30], which realizes the sharing of a secret S to n participants, while any k or more participants can recover the secret S. It mainly contains the following steps:

(1) Each participant U_i chooses a secret $a_0^{(i)}$ and constructs a $k-1$ order polynomial $f_i(x) = \sum_j = 0^{k-1} a_j^{(i)} x^j = a_0^{(i)} + a_1^{(i)} x + a_2^{(i)} x^2 + \cdots + a_{k-1}^{(i)} x^{k-1}$.

(2) Each participant U_i takes n numbers $x_1, x_2, \ldots, x_n$ and substitutes them into polynomials to get $s_j^{(i)} = f_i(x_j)$ $(j \in [1,n])$. The participant remains one of them and afterwards sends to the rest $n-1$ participants.

(3) After $U_j (j \in [1,n])$ receiving $s_j^{(i)}$ sent by the remaining $n-1$ participants $U_i (i \in [1,n], i \neq j)$, it is summed to get its share $s_j = \sum_{i=1}^{n} s_j^{(i)}$.

Any k or more participants can utilize the set of shares F to recover the secret S through the Lagrange interpolation formula $S = \sum_{j \in F} (s_j \prod_{i \in F, i \neq j} (i/(i-j)))$. By integrating these techniques into the basic BEvote, we propose a secure E-voting protocol, and Figure 4 shows its overall description. Firstly, Voters, Mixers, and Candidates use public key cryptography-based registration to realize the correct verification of their Bitcoin addresses. Secondly, Voter obtains an anonymous address through the identity confusion strategy. Thirdly, with the anonymous address, Voter in AVL executes joint Shamir random secret sharing to obtain secret shares and generate ballots with them. Fourthly, Voter constructs the voting transaction and broadcast it to the Bitcoin blockchain platform. Fifthly, each Candidates searches the voting transactions in blockchain according to *AVL* and collects the secret shares stored in the ballots if successfully decrypted them. If sufficient secret shares are collected, a winning transaction is constructed; otherwise, the Voter constructs a refund transaction.

4.2. Detailed Description of Proposed Protocol. There are five steps in our proposed protocol. The implementation details can be stated as follows:

In step 1, Voters, Candidates, and Mixers, respectively, make the registration to Supervisor.

(i) Each *Voter* submits the Bitcoin address BA_i, public key pu_i and identity information (including identity signature) to Supervisor. After Supervisor reviews the identity information and verifies that the signature corresponds to BA_i, it distributes a Voter's number and adds it to the private voter list.

(ii) Each Candidate submits the public key $PU_j = PR_j \cdot G$ and identity information to Supervisor, where PR_j is its private key and G is the basic point of elliptic curve. After being reviewed by Supervisor, the candidate list is constructed and published.

(iii) Each *Mixer* submits identity information to the regulator, and then, Supervisor issues a group certificate to it after it has passed the review.

In step 2, each Voter utilizes the identity confusion strategy to obtain the anonymous address. Then, the anonymous voter list is constructed.

In step 3, each Voter utilizes anonymous Bitcoin addresses to execute joint Shamir random secret sharing in order to obtain secret shares.

(i) Voter selects a secret value $a_0^{(i)}$ to construct a $k-1$ order polynomial $f_i(x) = \sum_{j=0}^{k-1} a_j^{(i)} x^j$.

(ii) Voter takes n numbers $x_j (j \in [1,n])$ and substitutes them with the polynomial. Then, the results $f_i(x_j)$ are sent to the remaining $(n-1)$ Voters. Each Voter needs to sum up the n data received to get the private key share $s_j = \sum_{i=1}^{n} f_i(x_j) (i \in [1,n])$. In

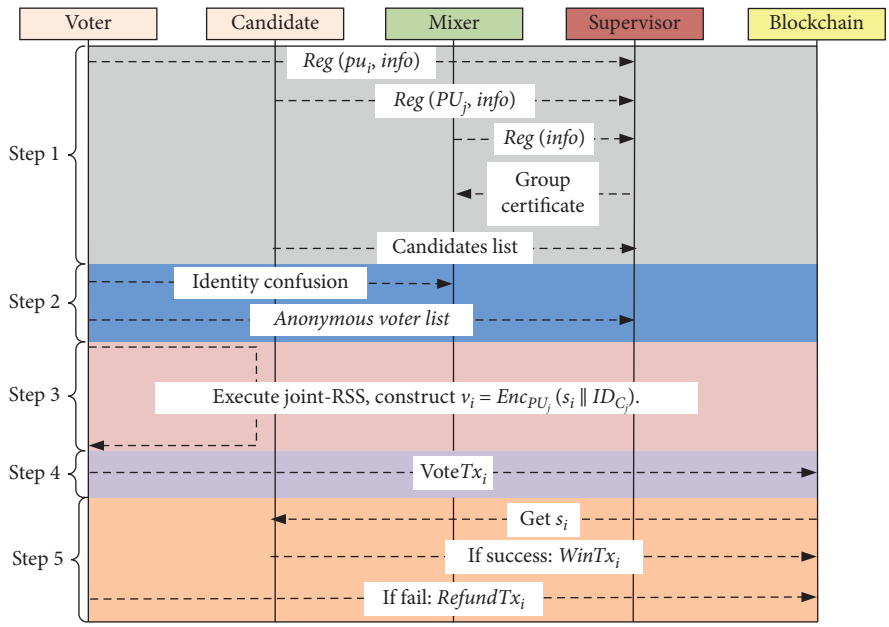

Figure 4: Secure E-voting protocol for BEvote.

order to construct the correct P2SH script to broadcast the voting information, the public key corresponding to this private key needs to be calculated. Then, the public key share $p_j = s_j \cdot G$ is calculated and sent it to other legitimate Voters.

(iii) When receiving the remaining $n-1$ public key shares p_i, a complete public key $P = \sum_{i=1}^{k} \lambda p_i = S \cdot G$ is constructed. Afterwards, each Voter utilizes the public key of his supported candidate C_j to encrypt the secret share s_i and ID_{C_j} to generate the ballot. For example, if a Voter U_i votes for a Candidate C_j, the form of the ballot is $v_i = \mathrm{Enc}_{PU_j}(s_i \parallel ID_{C_j})$.

In step 4, we need to put the secret sharing share in the ballot and change the first unlocking condition to secret shared private key. The redeem script RS of the first unlock condition contains the secret shared public key P and stores the ballot v_i. It can be unlocked by utilizing the corresponding secret shared private key signature. The form of this $1-1$ combination script is as follows:

$$1\langle P\rangle\ \langle v_i\rangle 1\ \mathrm{OP\_CHECKMULTISIG}. \quad (10)$$

In step 5, Candidates collect their own ballots to obtain secret shared shares. The Candidate C_j queries for VoteTx_i on the blockchain according to the transaction records of all addresses in AVL, checks the public ballot v_i in the P2SH script, and then tries to decrypt v_i with his private key PR_j in turn. Successful decryption can obtain the private key share s_i and the candidate identity number ID_{C_j}.

Depending on whether Candidate would collect enough secret shares contained in the ballots before timeout, there are two cases:

(i) *Voting Success*. If a Candidate C_j successfully collects k or more private key shares s_i, the secret shared private key $S = \sum_{i \in F}(s_i \prod_{j \in F, j \neq i}(j/(j-i)))$ can be reconstructed by the Lagrange interpolation formula. The Candidate can construct a winning transaction $WinTx_j$ for each voting transaction or use the output of multiple voting transactions as the input of a winning transaction. S is used to sign each input to unlock the first condition of the voting transaction output script. All UTXOs are obtained in n voting transactions. Other entities can verify that the *Can di date* has obtained sufficient secret shares, successfully reconstructed the private key, and won the voting by querying the winning transactions recorded in the blockchain. The input script of a winning transaction unlocks UTXO in the form of P2SH in the voting transaction. A signature of S and a serialized redeem script are required: $OP_0\langle \mathrm{Sig}_s\rangle\langle$Serialized Redeem Script$\rangle$.

(ii) *Voting Failed*. If no Candidate gets enough valid ballots to reconstruct the secret shared private key S, the voting fails. This voting protocol uses Bitcoin's time lock script to design a secure rollback mechanism. Even in the event of a vote failure, the Bitcoins submitted by Voters can still be redeemed. All Voters get Bitcoins back and vote again. Only when the time exceeds T, the CLTV operator of the second unlock condition would be invalidated. T denotes the agreed voting timeout time before voting. It can prevent voters from transferring Bitcoins before the failure. After that, each Voter constructs a refund transaction RefundTx_i and returns $n \cdot x$ Bitcoins to the original anonymous address ABA_i. This transaction input is the second condition of the voting transaction output script and is signed by the voter utilizing the private key.

5. Security Analysis

This section mainly analyzes the effectiveness and robustness of BEvote. In particular, we focus on proving its features of validity, robustness, anonymity, fairness, verifiability, and receipt-free.

Theorem 1. *(validity). Only valid entities reviewed can participate in the voting, which ensures the credibility of results.*

Proof. Before voting, *Voters* need to prove that they have enough Bitcoin for voting, and *Supervisors* will check the validity of the identity information of *Voters*, *Can di dates*, and *Mixers*. Although valid *Voters* obtained the *ABA* through the mix coin strategy while cutting off the physical connection between *ABA* and the real identity, *ABA* will be included in *AVL*. While in the pre-voting stage, only valid *Voters* in *AVL* can perform joint-RSS to obtain private key shares s_i to construct ballots. In the voting stage, only transactions constructed with addresses in *AVL* are considered valid.

Theorem 2. *(robustness). The system has certain fault tolerance to resist attacks and avoid errors.*

Proof. *Voters* who have not passed the review will not appear in doc missing and cannot participate in secret sharing to obtain shares. So, it is impossible to construct a valid ballot or launch an Sybil attack by generating multiple Bitcoin addresses. Ballots that are not constructed in accordance with the protocol's prescribed format, as well as ballots constructed using counterfeit shares, cannot reconstruct the key *S*. Hence, the ballots will be considered invalid. *Voters* may use the same shared secret to construct multiple ballots for the purpose of more voting rights, but the openness of the Bitcoin transaction script determines that such actions can be found and held accountable. This behavior also means that the number of bytes in the transaction script will increase, which will lead to an increase in transaction fees the *Voters* need to pay, so that it will be suppressed to a certain extent. Therefore, ballot forgery is in vain. In addition, external adversaries may launch denial-of-service attacks on system nodes to attempt to impact the voting process and results, yet it will be prevented by Bitcoin's distributed peer-to-peer network. Meanwhile, external adversaries will be spotted if they try to participate in voting by disguising themselves as system characters since their Bitcoin address has not been reviewed by Supervisor and is not on the public list.

Theorem 3. *(anonymity). Anonymity should be satisfied for the purpose of protecting the privacy of voters.*

Proof. This e-voting protocol achieves the purpose of sending and disclosing ballots by attaching voting data to transaction scripts. During the mix coin phase, the entity relationship between the anonymous Bitcoin address ABA_i and the voter U_i was severed. Therefore, after mix coin, *Voters* are anonymous during the execution of secret sharing and participation in Bitcoin transactions. At the same time, the voucher exchange method between multiple *Mixers* in the mix coin avoids the possibility of a single *Mixer* leaking user information.

Theorem 4. *(fairness). The decision of voters cannot be influenced by the fairness of voting protocol.*

Proof. Each *Voter*'s share of the private key s_i is only owned by himself before voting and is different from each other. The secret share s_i and candidate number ID_{C_j} in the ballot are encrypted by the public key of the selected candidate PU_j, and only the supported *Can di date* C_j has the private key PR_j to decrypt the ballot v_i. Due to the use of anonymous Bitcoin addresses, *Can di dates* cannot associate ballots with *Voters*' true identities. These designs make it impossible for *Voters* and *Can di dates* to know the current election situation or influence the decision of *Voters*, which ensures the fairness of voting.

Theorem 5. *(verifiability). Any entities could verify the voting process and the result even if the execution of protocol has ended.*

Proof. In our protocol, other entities can query the candidate list and anonymous voter list published by the regulator to verify the legitimacy of *Voters* and *Can di dates*. During the voting stage, ballots are recorded in transactions, and all transaction records are publicly available on the Bitcoin blockchain. Only if the *Can di date* collects enough secret shares to reconstruct the secret key, he can generate a winning transaction and withdraw BTCs. Other entities consult the transaction records stored in the blockchain, first verify that the serialized redeem script matches the hash value in UTXO, and then check whether the signature meets the multi-signature unlocking condition to verify that the *Can di date* has indeed enough ballots to reconstruct the key and win the vote.

Theorem 6. *(receipt-free). It means that the voting design needs to ensure that ballots that could not be proved in advance are sent by specific voters.*

Proof. This protocol cuts off the association between the anonymous Bitcoin address and the real identity during the mix coin phase; therefore, *Voters* cannot prove to a third party in advance that they have the right to vote. Meanwhile, during the voting stage, *Voters* need to execute joint-RSS to obtain shares. The share in the ballot is generated jointly by random parameters sent by all *Voters*. There is no way to prove to a third party that a specific ballot will be generated. These measures make it impossible for *Voters* to provide proof of voting to a third party in advance and guarantee the protocol receipt-free.

6. Performance Evaluation

In this section, we mainly evaluate the efficiency of BEvote. We first use the theoretical analysis to look at its computation and communication overhead. Then, we build the simulation platform to perform extensive experiments.

6.1. Theoretical Analysis. In BEvote, there are three key stages to affect its efficiency, including confusing *Voter*'s identity, generating ballots, and constructing voting transaction. Considering that BEvote utilizes the Bitcoin blockchain platform as the storage infrastructure, the efficiency on the third stage depends on the throughput of such blockchain. It is widely known that the rapid development of Bitcoin lightning network has significantly enhanced its transaction speed [24], and thus, the performance evaluation regarding the voting transaction construction is not necessary. In this case, we focus on investigating the efficiencies of the first two. When voters need to confuse their identities or generate ballots, the main execution is the identity confusion strategy and joint-RSS. Thus, we, respectively, evaluate their efficiencies [31].

The *Voter*'s identity confusion strategy contains three parts: *Mixer* registration, identity confusion, and confusion audit. Considering that the modular multiplication operation m and the modular exponentiation operation M are two time-consuming operations, we mainly focus on analyzing the number of such operations in each part. Table 3 shows the theoretical analysis results. Apparently, most of the computational overhead is concentrated in the confusion phase. The reason is that both *Voters* and *Mixers* need to calculate group blind signatures and RSA signatures.

We utilize the amount of data transmitted in the ballot generation stage to estimate the communication complexity. With regard to joint-RSS, each *Voter* is required to transmit a bits to the rest $n-1$ voters. Afterwards, each participant calculates b bits $p_i = s_i \cdot G$ and broadcasts it. As a result, there are $n \cdot a(n-1)$ bit transmission and $n \cdot b$ bit broadcast overhead. The communication overhead estimation is listed in Table 4. The $b + a(n-1)$ bit overhead is affordable for each one, while the total $n(b + a(n-1))$ bit overhead is trivial to the current network.

6.2. Experimental Analysis. The simulation utilizes the Java language, with the help of the JPBC library to implement the operations in the elliptic curve group and utilize multiple threads to simulate multiple voters [32]. The simulation environment is Windows 10 64 bit operating system with i7-8750H (@2.2 GHz) CPU and 16 GB memory. The process of program execution is that the user randomly constructs a polynomial by taking random numbers, then substitutes n numbers to evaluate the polynomial, and sends it to other users. Each user sums all the values they receive as their secret share. The following experimental values are all averaged after running 100 times in the Eclipse IDE [33].

To further proceed to investigate the execution time of voter's identity confusion, we, respectively, simulate the modular multiplication and modular exponentiation operations in each part. Figure 5 shows the simulation results. Since the computation complexity of modular exponentiation is far greater than modular multiplication, the

Table 3: Theoretical execution overhead of voter's identity confusion.

Part	Modular multiplication	Modular exponentiation
Mixer registration	0	$2n \cdot M$
Identity confusion	$4n \cdot m$	$21n \cdot M$
Confusion audit	$2n \cdot m$	$n \cdot M$

Table 4: Theoretical communication overhead of ballot generation.

Joint-RSS	Broadcast	Transmission	Sum
Individual	b	$a(n-1)$	$b + a(n-1)$
Total	$n \cdot b$	$n \cdot a(n-1)$	$n(b + a(n-1))$

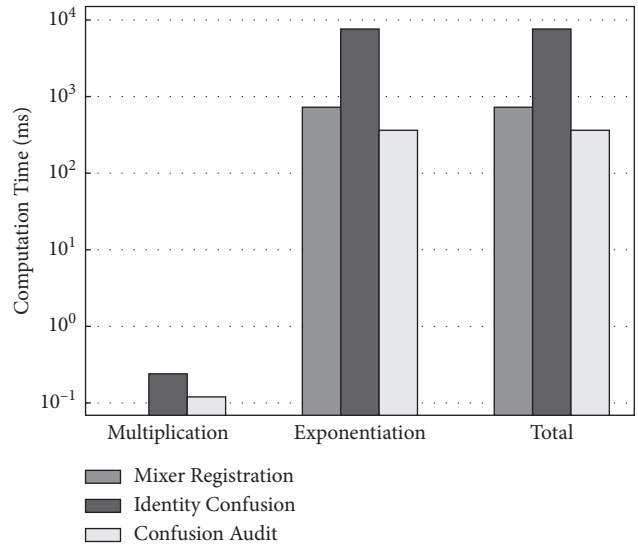

Figure 5: Execution time of voter's identity confusion.

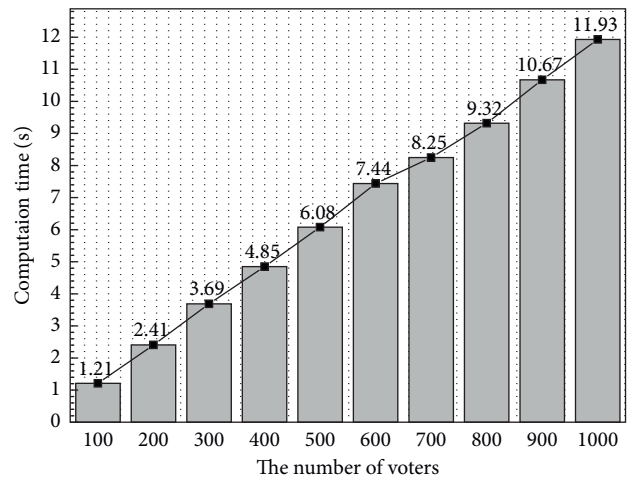

Figure 6: Execution time of ballot collection.

total execution time of identity confusion part is more than ten times as much as *Mixer* registration and confusion audit.

In addition, we consider the running time of ballot collection with the number of *Voters* from 0 to 1000. The results are shown in Figure 6. Apparently, the time overhead increases roughly linearly with the number of *Voters*. The computation time for the scale of thousands of *Voters* is less than one minute, so the time overhead for the large-scale system is within an acceptable range.

7. Conclusion

In this study, we propose a Bitcoin-based electronic voting scheme with anonymity and robustness, which abstracts a Bitcoin transaction as the voting process. Its characteristics can be summarized as follows: firstly, we design a coin mixing-based system model to achieve strong anonymity; secondly, we devise a secret sharing-based E-voting protocol to keep the voting numbers private; and finally, we carry out security analysis and experimental evaluations to demonstrate its efficiency and robustness.

There are many directions in which this work can be extended. The first is to consider the low throughput and high network delay of Bitcoin blockchain, which would cause the scalable issue in regional collaborative medical system. A promising solution is to directly utilize MEC nodes to build a private blockchain towards E-voting, instead of public blockchain. In this case, we can make this blockchain to be more voting-compatible, with both scalability and robustness. Another direction is to study the new applications of E-voting, such as decision support or recommendation system [34–36].

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Disclosure

Xin Xu is the co-first author of this work.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 62072092, 62072093, and U1708262); the China Postdoctoral Science Foundation (No. 2019M653568); the Key Research and Development Project of Hebei Province (No. 20310702D); the Natural Science Foundation of Hebei Province (No. F2020501013); and the Fundamental Research Funds for the Central Universities (No. N2023020).

References

[1] Y. Xu, H. Gao, and R. Li, Y. Yin, Z. Cao, and Z. Mai, QoS prediction for service recommendation with features learning in mobile edge computing environment," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1135–1145, 2020.

[2] M. Cao, J. Xiao, Q. Xu, H. Gao, K. Xu, and Y. Yin, "The deep features and attention mechanism based method to dish healthcare under social IoT systems: an empirical study with a hand-deep local-global net," *IEEE Transactions on Computational Social Systems*, pp. 1–12, 2021.

[3] M. Blaze, J. Braun, and C. G. Advisors, "Defcon 25 voting machine hacking village," in *Proceedings of the DEFCON*, pp. 1–18, Washington, DC, USA, September 2017.

[4] M. A. Specter, J. Koppel, and D. Weitzner, "The ballot is busted before the blockchain: a security analysis of voatz, the first internet voting application used in us federal elections," in *Proceedings of the 29th {USENIX} Security Symposium ({USENIX} Security 20)*, pp. 1535–1553, Anaheim, CA, USA, 2020.

[5] T. Moura and A. Gomes, "Blockchain voting and its effects on election transparency and voter confidence," in *Proceedings of the 18th Annual International Conference on Digital Government Research*, pp. 574-575, Staten Island, NY, USA, June 2017.

[6] J. Zhang, S. Zhong, T. Wang, H.-C. Chao, and J. Wang, ""Blockchain-based systems and applications: a survey," *Journal of Internet Technology*, vol. 21, no. 1, pp. 1–14, 2020.

[7] Z. Zhao and T.-H. H. Chan, "How to vote privately using bitcoin," in *Proceedings of the International Conference on Information and Communications Security*, pp. 82–96, Beijing, China, December 2015.

[8] C. K. Adiputra, R. Hjort, and H. Sato, "A proposal of blockchain-based electronic voting system," in *Proceedings of the 2018 Second World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pp. 22–27, IEEE, London, UK, October 2018.

[9] P. C. Jason and K. Yuichi, "E-voting system based on the bitcoin protocol and blind signatures," *IPSJ Transactions on Mathematical Modeling and its Applications*, vol. 10, no. 1, pp. 14–22, 2017.

[10] S. Bistarelli, M. Mantilacci, P. Santancini, and F. Santini, "An end-to-end voting-system based on bitcoin," in *Proceedings of the Symposium on Applied Computing*, pp. 1836–1841, Marrakech, Morocco, April 2017.

[11] S. Bartolucci, P. Bernat, and D. Joseph, "Sharvot: secret share-based voting on the blockchain," in *Proceedings of the 1st International Workshop on Emerging Trends in Software Engineering for Blockchain*, pp. 30–34, Gothenburg, Sweden, March 2018.

[12] Y. Takabatake, D. Kotani, and Y. Okabe, "An anonymous distributed electronic voting system using zerocoin," Technical Report, IEICE, 2019.

[13] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, 1981.

[14] B. Adida, "Helios: web-based open-audit voting," in *Proceedings of the USENIX Security Symposium*, vol. 17, pp. 335–348, California, CA, USA,, July 2008.

[15] N. Chondros, B. Zhang, T. Zacharias et al., "Distributed, end-to-end verifiable, and privacy-preserving internet voting systems," *Computers & Security*, vol. 83, pp. 268–299, 2019.

[16] V. Cortier, D. Galindo, R. Küsters, J. Mueller, and T. Truderung, "Sok: verifiability notions for e-voting protocols," in *Proceedings of the 2016 IEEE Symposium on Security and Privacy (SP)*, pp. 779–798, IEEE, San Jose, CA, USA, August 2016.

[17] J. Benaloh and D. Tuinstra, "Receipt-free secret-ballot elections," in *Proceedings of the ACM Symposium on the Theory of Computing (STOC)*, vol. 94, pp. 544–553, Montreal, PQ, Canada, May 1994.

[18] M. Andrychowicz, S. Dziembowski, D. Malinowski, and L. Mazurek, "Secure multiparty computations on bitcoin," in *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, pp. 443–458, IEEE, San Jose, CA, USA, May 2014.

[19] D. Chaum, C. Crépeau, and I. Damgard, "Multiparty unconditionally secure protocols," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, pp. 11–19, Chicago, IL, USA, May 1988.

[20] P. McCorry, S. F. Shahandashti, and F. Hao, "A smart contract for boardroom voting with maximum voter privacy," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, pp. 357–375, Springer, Sliema, Malta, January 2017.

[21] P. Tarasov and H. Tewari, "The future of e-voting." *IADIS International Journal on Computer Science & Information Systems*, vol. 12, no. 2, 2017.

[22] E. Yavuz, A. K. Koç, U. C. Çabuk, and G. Dalkılıç, "Towards secure E-voting using ethereum blockchain," in *Proceedings of the 2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, pp. 1–7, IEEE, Antalya, Turkey, March 2018.

[23] B. Shahzad and J. Crowcroft, "Trustworthy electronic voting using adjusted blockchain technology," *IEEE Access*, vol. 7, pp. 24477–24488, 2019.

[24] J. Poon and T. Dryja, *The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments*, 2016, https://lightning.network/lightningnetwork-paper.pdf.

[25] S. Zhang, L. Wang, and H. Xiong, "Chaintegrity: blockchain-enabled large-scale e-voting system with robustness and universal verifiability," *International Journal of Information Security*, vol. 19, no. 3, pp. 1–19, 2019.

[26] N. Lu, Y. Chang, W. Shi, and K.-K. Raymond Choo, "CoinLayering: an efficient coin mixing scheme for large scale bitcoin transactions," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–17, 2020.

[27] T. Fei, Y. Chang, J. Wang, N. Lu, and W. Shi, "Anonymous bitcoin mixing scheme based on semi-trusted supervisor," in *Proceedings of the 2020 IEEE 3rd International Conference on Electronics Technology (ICET)*, pp. 845–850, Chengdu, China, May 2020.

[28] A. Fiat and A. Shamir, "How to prove yourself: practical solutions to identification and signature problems," in *Proceedings of the Advances in Cryptology–CRYPTO'86*, Santa Barbara, CA, USA, December 2000.

[29] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," Technical Report, Manubot, 2019.

[30] Z. Chen, S. Li, Q. Huang, J. Yan, and Y. Ding, "A joint random secret sharing scheme with public verifiability," *International Journal on Network Security*, vol. 18, no. 5, pp. 917–925, 2016.

[31] K. M. Khan, J. Arshad, and M. M. Khan, "Investigating performance constraints for blockchain based secure e-voting system," *Future Generation Computer Systems*, vol. 105, pp. 13–26, 2020.

[32] A. De Caro and V. Iovino, "JPBC: Java pairing based cryptography," in *Proceedings of the 16th IEEE symposium on computers and communications*, pp. 850–855, IEEE, Kerkyra, Corfu, Greece, July 2011.

[33] E. Foundation, "Enabling open innovation collaboration. The eclipse foundation," 2021, https://www.eclipse.org.

[34] H. Gao, X. Qin, R. J. D. Barroso, W. Hussain, Y. Xu, and Y. Yin, "Collaborative learning-based industrial IoT API recommendation for software-defined devices: the implicit knowledge discovery perspective," *IEEE Transactions on Emerging Topics in Computational Intelligence*, pp. 1–11, 2020.

[35] M. Cao, X. Yang, and S. Zhou, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *ACM/Springer Mobile Networks and Applications*, vol. 25, no. 2, pp. 376–390, 2020.

[36] H. Gao, W. Hussain, Y. Huang, and H. Xu, "Ssur: an approach to optimizing virtual machine allocation strategy based on user requirements for cloud data center," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 2, pp. 370–381, 2021.